Analysis of rounding error accumulation in Conjugate Gradients to improve the maximal attainable accuracy of pipelined CG

Siegfried Cools, Emrullah Fatih Yetkin, Emmanuel Agullo, Luc Giraud, Wim Vanroose

Analysis of rounding error accumulation in Conjugate Gradients to improve the maximal attainable accuracy of pipelined CG

Siegfried Cools, Emrullah Fatih Yetkin, Emmanuel Agullo, Luc Giraud, Wim Vanroose

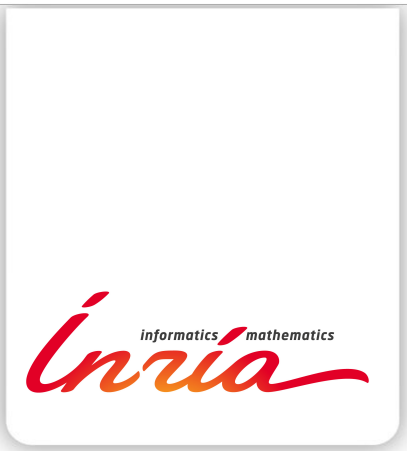
RESEARCH REPORT

N° 8849

January 2016

Project-Teams HiePACS

ISSN 0249-6399 ISRN INRIA/RR--8849--FR+ENG



Analysis of rounding error accumulation in Conjugate Gradients to improve the maximal attainable accuracy of pipelined CG

Siegfried Cools*, Emrullah Fatih Yetkin†, Emmanuel Agullo†,
Luc Giraud†, Wim Vanroose*

Project-Teams HiePACS

Research Report n° 8849 — January 2016 — 27 pages

Abstract: Pipelined Krylov solvers typically offer improved strong scaling on parallel HPC hardware compared to standard Krylov methods for large and sparse linear systems. In pipelined Krylov algorithms the traditional synchronization bottleneck is mitigated by overlapping time-consuming global communications with useful computations. However, to achieve this communication hiding strategy, pipelined methods introduce additional recurrence relations for a number of auxiliary variables that are required to update the guess for the solution. This paper aims at studying the influence of rounding errors on the convergence of the pipelined Conjugate Gradient (CG) method. We explain why roundoff effects have a significantly larger impact on the accuracy and convergence of the pipelined CG algorithm in comparison to the traditional CG method. Furthermore, a model for the gap between the true residual and the recursively computed residual used in the algorithm is proposed. Based on this error model we suggest an automated residual replacement strategy to reduce the effect of rounding errors on the final iterative solution. The resulting pipelined CG method with residual replacement improves the maximal attainable accuracy of pipelined CG, while maintaining the efficient parallel performance of the pipelined method. This conclusion is substantiated by numerical results for a variety of benchmark problems.

Key-words: Conjugate gradients, Parallelization, Latency hiding, Pipelining, Rounding errors, Maximal attainable accuracy, Global communication

* Applied Mathematics Group, University of Antwerp, Middelheimlaan 1, 2020 Antwerp, BE
† HiePACS, Inria Bordeaux - Sud-Ouest, 200 Avenue de la Vieille Tour, 33405 Talence, FR

RESEARCH CENTRE
BORDEAUX – SUD-OUEST

200, Avenue de la vieille tour
33405 Talence Cedex

Analyse de l'accumulation d'erreurs d'arrondis dans la Gradient COnjugué pour améliorer la précision attaignable de sa version pipelinée

Résumé : Les méthodes de Krylov pipelinées offrent généralement de meilleures performance en terme de passage à l'échelle parallèle. Nous analysons dans ce travail l'effet de l'arithmétique finie sur la précision atteinte par la variante pipelinée du Gradient conjugué et proposons une alternative pour l'améliorer.

Mots-clés : Gradient conjugué, parallélisation, masquer la latence, erreurs d'arrondis, meilleure précision atteignable, communication globale

Contents

1 Introduction

Krylov subspace methods [24] form the basis linear algebra solvers for many contemporary high-performance computing applications. The Conjugate Gradient (CG) method, which dates back to the 1952 paper by Hestenes and Stiefel [22], can be considered as the first of these Krylov methods. Although over 60 years old, the CG method is still the work horse method for the solution of linear systems with symmetric positive definite (SPD) matrices due to its numerical simplicity and easy implementation. These SPD systems originate from various applications, such as the simulation of problems modeled by a partial differential equation (PDE).

Due to the transition of hardware towards the exascale regime in the coming years, research on the scalability of Krylov methods on massively parallel architectures has become increasingly prominent. This is reflected in the new High Performance Conjugate Gradients (HPCG) benchmark for ranking HPC systems introduced by Dongarra et al. in 2013 [12, 13]. The ranking is based on sparse matrix-vector computations and data access patterns, rather than the dense matrix algebra used in the traditional High Performance LINPACK (HPL) benchmark. Moreover, since the system matrix is often sparse, the main bottleneck for efficient parallel execution is typically not the sparse matrix-vector product (SPMV), but the communication overhead (bandwidth saturation) caused by global reductions required in the computation of dot-products.

Recently significant research has been devoted to the reduction and/or elimination of the synchronization bottleneck in Krylov methods. The earliest papers on synchronization reduction and latency hiding in Krylov methods date back to the early 1990's, see [2, 8, 11, 15]. The idea of reducing the number of global communication points in Krylov methods on parallel computer architectures was also used in the s-step methods by Chronopoulos et al. [5, 6, 7] and more recently by Carson et al. in [3, 4]. In addition to communication avoiding methods, research on hiding global communication by overlapping communication with computations was performed by a various authors over the last decades, see Demmel et al. [11], De Sturler et al. [9], and Ghysels et al. [16, 17].

The pipelined CG (p-CG) method proposed in [17] aims at hiding the global synchronization latency of standard preconditioned CG by removing some of the costly global synchronization points. Pipelined CG performs only one global reduction per iteration. Furthermore, this global communication phase is overlapped by the sparse matrix-vector product (SPMV), which requires only local communication. In this way, idle core time is minimized by performing useful computations simultaneously to the expensive global communication phase, cf. [14].

The reorganization of the CG algorithm that is performed to achieve the overlap of communication with computations introduces several additional AXPY ($y \leftarrow \alpha x + y$) operations to recursively compute auxiliary variables. Vector operations such as an AXPY are typically computed locally, and thus do not require communication between nodes. Thus, the addition of extra recurrences has no impact on the communication flow of the algorithm. Dot-products of two vectors, on the other hand, involve global communication between all processes, and are therefor grouped together in p-CG.

In exact arithmetic, the resulting pipelined CG algorithm is equivalent to standard CG. However, when switching to finite precision, each of the additional recurrences introduce local rounding errors. The propagation of these rounding errors throughout the algorithm is much more pronounced for pipelined CG, and can have a detrimental effect on the convergence of the algorithm. As a result, a significant loss of attainable accuracy and a delayed convergence compared to classic CG can in practice be observed for the p-CG algorithm. The current paper focuses on analyzing the rounding error propagation in different variants of the CG algorithm. Additionally, the analytical results will be used to formulate an automated residual replacement strategy that counteracts the propagation of rounding errors, and hence improves the maximal

attainable accuracy of the pipelined CG method.

The paper is structured as follows. In Section 2 the propagation of rounding errors in standard preconditioned CG, Chronopoulos/Gear CG, and the pipelined CG algorithm is analyzed. Bounds for the gap between the explicitly computed residual and the recursive residual are derived. Section 3 proposes an approximate and practically useable model for the residual gap. Furthermore, the incorporation of a residual replacement strategy in the pipelined CG method is discussed. A criterion for automated residual replacement based on the aforementioned error propagation model is suggested. Extensive numerical experiments in Section 4 show the validity of the error model and the possible improvement in attainable accuracy for the pipelined CG method with automated residual replacement. We illustrate that the residual replacement strategy does not affect the parallel scalability of the pipelined CG method. Finally, conclusions are formulated in Section 5.

2 Analysis of rounding error propagation in CG algorithms

The analysis in this section is based upon the rounding error analysis performed by Greenbaum in [19] and Strakoš & Gutknecht in [20]. Additional work on this topic can be found in [10, 18, 25, 27, 30, 34]. We assume the following classical model for floating point arithmetic on a machine with machine precision ϵ:

$$\mathrm{fl}(a \pm b) = a(1+\epsilon_1) \pm b(1+\epsilon_2), \quad |\epsilon_1|, |\epsilon_2| \leq \epsilon, \tag{1}$$

$$\mathrm{fl}(a \ \mathrm{op}\ b) = (a \ \mathrm{op}\ b)(1+\epsilon_3), \quad |\epsilon_3| \leq \epsilon, \quad \mathrm{op} = *, /. \tag{2}$$

Under this model, and discarding terms involving ϵ^2 or higher powers of ϵ when terms of order ϵ are present, the following standard results for operations on an n-by-n matrix A, n-length vectors v and w and scalar number α hold:

$$\|\alpha v - \mathrm{fl}(\alpha v)\| \leq \|\alpha v\|\,\epsilon = |\alpha|\,\|v\|\,\epsilon, \tag{3}$$

$$\|v + w - \mathrm{fl}(v+w)\| \leq (\|v\| + \|w\|)\,\epsilon, \tag{4}$$

$$|\,(v,w) - \mathrm{fl}(\,(v,w)\,)| \leq n\,\|v\|\,\|w\|\epsilon, \tag{5}$$

$$\|Av - \mathrm{fl}(Av)\| \leq (m\sqrt{n})\,\|A\|\,\|v\|\,\epsilon, \tag{6}$$

where m is the maximum number of nonzeros in any row of A. The norm $\|\cdot\|$ denotes the Euclidean 2-norm throughout this manuscript, unless explicitly stated otherwise.

2.1 Propagation of rounding errors in classical CG

Classical preconditioned CG is given by Algorithm 1. Note that in the unpreconditioned case, line 8 is dropped, and each occurrence of u_i is replaced by r_i. In finite precision arithmetic, the recurrences for the search directions p_i, iterates x_i and residuals r_i in iteration i ($i = 0, 1, 2, \ldots$) of the CG algorithm have to be replaced by

$$\begin{aligned} p_{i+1} &= u_{i+1} + \beta_{i+1} p_i + \delta_i^p, \\ x_{i+1} &= x_i + \alpha_i p_i + \delta_i^x, \\ r_{i+1} &= r_i - \alpha_i A p_i + \delta_i^r, \end{aligned} \tag{7}$$

Algorithm 1 Preconditioned CG

1: **procedure** PREC-CG(A, M^{-1}, b, x_0)
2: $r_0 := b - Ax_0$; $u_0 := M^{-1}r_0$; $p_0 = u_0$
3: **for** $i = 0, \ldots$ **do**
4: $s_i := Ap_i$
5: $\alpha_i := (r_i, u_i) \,/\, (s_i, p_i)$
6: $x_{i+1} := x_i + \alpha_i p_i$
7: $r_{i+1} := r_i - \alpha_i s_i$
8: $u_{i+1} := M^{-1} r_{i+1}$
9: $\beta_{i+1} := (r_{i+1}, u_{i+1}) \,/\, (r_i, u_i)$
10: $p_{i+1} := u_{i+1} + \beta_{i+1} p_i$
11: **end for**
12: **end procedure**

where δ_i^p, δ_i^r and δ_i^x contain the local rounding errors produced in step i, and p_i, r_i and x_i denote the actually computed quantities. By $r_i = b - Ax_i$ we will always denote the computed quantity. This should cause no confusion since the quantities generated in exact arithmetic are unavailable in practice. Vectors computed by actually applying the matrix-vector product will be called *explicit* quantities, in contrast to the *recursive* quantities given by (7). Since the SPMV is a computationally costly operation, the explicit r_i (also commonly denoted 'true residual' in this context) is replaced by its recursive definition in Algorithm 1.

The iteration i rounding errors satisfy the following bounds:

$$\begin{aligned} \|\delta_i^p\| &\leq (\|u_{i+1}\| + 2\,|\beta_{i+1}|\,\|p_i\|)\,\epsilon, \\ \|\delta_i^x\| &\leq (\|x_i\| + 2\,|\alpha_i|\,\|p_i\|)\,\epsilon, \\ \|\delta_i^r\| &\leq \left(\|r_i\| + (m\sqrt{n}+2)\,|\alpha_i|\,\|A\|\,\|p_i\|\right)\epsilon, \end{aligned} \tag{8}$$

We now want to estimate the difference (or gap) between the true residuals $b - Ax_i$ and the recursive residuals r_i. Hence, we define this gap as

$$f_i = (b - Ax_i) - r_i. \tag{9}$$

For $i = 0$, the residual r_0 is computed explicitly in Algorithm 1, and the gap f_0 is thus the roundoff from computing r_0 from A, x_0 and b, i.e. $f_0 = b - Ax_0 - \mathrm{fl}(b - Ax_0)$. The norm of this initial gap is bounded by

$$\|f_0\| \leq \left((m\sqrt{n}+1)\,\|A\|\,\|x_0\| + \|b\|\right)\epsilon. \tag{10}$$

In iteration i we obtain the following formula for the gap by substituting the recursions (7):

$$\begin{aligned} f_{i+1} &= (b - Ax_{i+1}) - r_{i+1} \\ &= b - A(x_i + \alpha_i p_i + \delta_i^x) - (r_i - \alpha_i A p_i + \delta_i^r) \\ &= f_i - A\delta_i^x - \delta_i^r. \end{aligned} \tag{11}$$

This recursive formulation relates the residual error f_{i+1} in step i to the previous residual error f_i. By taking norms on both sides, we obtain an upper bound on $\|f_{i+1}\|$ in function of the previous gap $\|f_i\|$:

$$\|f_{i+1}\| \leq \|f_i\| + \|A\delta_i^x + \delta_i^r\| \tag{12}$$

where we can use the bounds (8) to further rewrite the right-hand side bound as

$$\begin{aligned}\|A\delta_i^x + \delta_i^r\| &\leq \|A\|\,\|\delta_i^x\| + \|\delta_i^r\| \\ &\leq \left(\|A\|\,\|x_i\| + 2\,|\alpha_i|\,\|A\|\,\|p_i\| + \|r_i\| + (m\sqrt{n}+2)\,|\alpha_i|\,\|A\|\,\|p_i\|\right)\epsilon \\ &= \left(\|A\|\,\|x_i\| + (m\sqrt{n}+4)\,|\alpha_i|\,\|A\|\,\|p_i\| + \|r_i\|\right)\epsilon \\ &:= e_i^f\,\epsilon. \end{aligned} \tag{13}$$

Hence, with the above definition of the upper bound factor e_i^f, we obtain

$$\|f_{i+1}\| \leq \|f_i\| + e_i^f\,\epsilon, \tag{14}$$

which gives an upper bound on the norm of the gap between the true and recursive residual in any iteration of the method. The recurrence (11) implies that in the classical CG method the gap after $i+1$ iterations is simply the sum of local rounding errors; see also [19, 20].

2.2 Propagation of rounding errors in Chronopoulos/Gear CG

In so-called Chronopoulos/Gear CG (commonly denoted CG-CG in this manuscript), Algorithm 2, an extra recurrence for the auxiliary variable $s_i = Ap_i$ is introduced, and the auxiliary vectors $w_i = Au_i$ and $u_i = M^{-1}r_i$ are computed explicitly in each iteration. Alg. 2 avoids communication by reducing the two global reduction phases of classic CG to one global synchronization (lines 11-12). Restriction to the unpreconditioned algorithm can be achieved by simply removing line 9 and replacing u_i by r_i where required. In inexact arithmetic the corresponding recurrences in Alg. 2 are replaced by

$$\begin{aligned} x_{i+1} &= x_i + \alpha_i p_i + \delta_i^x, & p_i &= u_i + \beta_i p_{i-1} + \delta_i^p, \\ r_{i+1} &= r_i - \alpha_i s_i + \delta_i^r, & s_i &= Au_i + \beta_i s_{i-1} + \delta_i^s, \end{aligned} \tag{15}$$

where the local rounding errors satisfy

$$\begin{aligned}\|\delta_i^x\| &\leq (\|x_i\| + 2\,|\alpha_i|\,\|p_i\|)\,\epsilon, \\ \|\delta_i^r\| &\leq (\|r_i\| + 2\,|\alpha_i|\,\|s_i\|)\,\epsilon, \\ \|\delta_i^p\| &\leq \left((\tilde{m}\sqrt{n}+1)\,\|M^{-1}\|\,\|r_i\| + 2\,|\beta_i|\,\|p_{i-1}\|\right)\epsilon, \\ \|\delta_i^s\| &\leq \left((m\sqrt{n}+\tilde{m}\sqrt{n}+1)\,\|A\|\,\|M^{-1}\|\,\|r_i\| + 2\,|\beta_i|\,\|s_{i-1}\|\right)\epsilon, \end{aligned} \tag{16}$$

where $\tilde{m}$ is the maximum number of nonzeros in any row of the operator M^{-1}. To estimate the gap between the true and recursive residual we again substitute the recursions (15) in $f_i = (b - Ax_i) - r_i$. Note that we have

$$\|f_0\| \leq \left((m\sqrt{n}+1)\,\|A\|\,\|x_0\| + \|b\|\right)\epsilon, \tag{17}$$

since the recursion for x_i is identical to that in CG, see (7). The gap in iteration i is given by

$$\begin{aligned} f_{i+1} &= (b - Ax_{i+1}) - r_{i+1} \\ &= b - A(x_i + \alpha_i p_i + \delta_i^x) - (r_i - \alpha_i s_i + \delta_i^r) \\ &= f_i - \alpha_i g_i - A\delta_i^x - \delta_i^r, \end{aligned} \tag{18}$$

where $g_i = Ap_i - s_i$, that is the gap between the explicit and recursively computed auxiliary variable s_i. For the latter gap, it holds for $i = 0$ that

$$\|g_0\| \leq m\sqrt{n}\,\|A\|\,\|p_0\|\,\epsilon. \tag{19}$$

Algorithm 2 Preconditioned Chronopoulos/Gear CG

1: **procedure** PREC-CG-CG(A, M^{-1}, b, x_0)
2: $r_0 := b - Ax_0$; $u_0 := M^{-1}r_0$; $w_0 := Au_0$
3: $\alpha_0 := (r_0, u_0)/(w_0, u_0)$; $\beta_0 := 0$; $\gamma_0 := (r_0, u_0)$
4: **for** $i = 0, \ldots$ **do**
5: $p_i := u_i + \beta_i p_{i-1}$
6: $s_i := w_i + \beta_i s_{i-1}$
7: $x_{i+1} := x_i + \alpha_i p_i$
8: $r_{i+1} := r_i - \alpha_i s_i$
9: $u_{i+1} := M^{-1} r_{i+1}$
10: $w_{i+1} := Au_{i+1}$
11: $\gamma_{i+1} := (r_{i+1}, u_{i+1})$
12: $\delta := (w_{i+1}, u_{i+1})$
13: $\beta_{i+1} := \gamma_{i+1}/\gamma_i$
14: $\alpha_{i+1} := (\delta/\gamma_{i+1} - \beta_{i+1}/\alpha_i)^{-1}$
15: **end for**
16: **end procedure**

Indeed, in iteration $i = 0$, the error is the norm of the roundoff on the explicit computation $s_0 = Ap_0$. In iteration i the variable s_i is computed recursively, and it holds that

$$\begin{aligned} g_i &= Ap_i - s_i \\ &= A(u_i + \beta_i p_{i-1} + \delta_i^p) - (Au_i + \beta_i s_{i-1} + \delta_i^s) \\ &= \beta_i g_{i-1} + A\delta_i^p - \delta_i^s. \end{aligned} \tag{20}$$

The residual gap estimate is thus given by the coupled recursions

$$\begin{bmatrix} f_{i+1} \\ g_i \end{bmatrix} = \begin{bmatrix} 1 & -\alpha_i\beta_i \\ 0 & \beta_i \end{bmatrix} \begin{bmatrix} f_i \\ g_{i-1} \end{bmatrix} + \begin{bmatrix} -A\delta_i^x - \delta_i^r - \alpha_i\left(A\delta_i^p - \delta_i^s\right) \\ A\delta_i^p - \delta_i^s \end{bmatrix}. \tag{21}$$

Taking norms, we obtain the upper bounds

$$\begin{bmatrix} \|f_{i+1}\| \\ \|g_i\| \end{bmatrix} \leq \begin{bmatrix} 1 & |\alpha_i\beta_i| \\ 0 & |\beta_i| \end{bmatrix} \begin{bmatrix} \|f_i\| \\ \|g_{i-1}\| \end{bmatrix} + \begin{bmatrix} \|A\delta_i^x + \delta_i^r\| + |\alpha_i|\|A\delta_i^p - \delta_i^s\| \\ \|A\delta_i^p - \delta_i^s\| \end{bmatrix}. \tag{22}$$

This bound can be further rewritten into more tractable expressions using the bounds for the local rounding errors in (16), i.e.

$$\begin{aligned} \|A\delta_i^x + \delta_i^r\| &\leq \|A\|\,\|\delta_i^x\| + \|\delta_i^r\| \\ &\leq \left(\|A\|\,\|x_i\| + 2\,|\alpha_i|\,\|A\|\,\|p_i\| + \|r_i\| + 2\,|\alpha_i|\,\|s_i\|\right)\epsilon \\ &:= e_i^f \epsilon, \end{aligned} \tag{23}$$

and

$$\begin{aligned} \|A\delta_i^p - \delta_i^s\| &\leq \|A\|\,\|\delta_i^p\| + \|\delta_i^s\| \\ &\leq \big((\tilde{m}\sqrt{n}+1)\,\|A\|\,\|M^{-1}\|\,\|r_i\| + 2\,|\beta_i|\,\|A\|\,\|p_{i-1}\| \\ &\quad + (m\sqrt{n} + \tilde{m}\sqrt{n} + 1)\,\|A\|\,\|M^{-1}\|\,\|r_i\| + 2\,|\beta_i|\,\|s_{i-1}\|\big)\,\epsilon \\ &= \big(2\,|\beta_i|\,\|A\|\,\|p_{i-1}\| + ((m+2\tilde{m})\sqrt{n}+2)\,\|A\|\,\|M^{-1}\|\,\|r_i\| \\ &\quad + 2\,|\beta_i|\,\|s_{i-1}\|\big)\,\epsilon \\ &:= e_i^g \epsilon. \end{aligned} \tag{24}$$

Note that the definitions of the bounds are local to each subsection; definition (23) above holds for CG-CG, and should not be confused with the earlier identical notation defined by (13) for the bound in classical CG. Hence, with the factors e_i^f and e_i^g defined as above, the norm of the gap between the true and recursive residual is bounded by the recursively defined system of upper bounds

$$\begin{bmatrix} \|f_{i+1}\| \\ \|g_i\| \end{bmatrix} \leq \begin{bmatrix} 1 & |\alpha_i\beta_i| \\ 0 & |\beta_i| \end{bmatrix} \begin{bmatrix} \|f_i\| \\ \|g_{i-1}\| \end{bmatrix} + \begin{bmatrix} e_i^f\epsilon + |\alpha_i|\, e_i^g\epsilon \\ e_i^g\epsilon \end{bmatrix}. \tag{25}$$

Note that this is in sharp contrast to the error behavior of the residual gap in the classical CG algorithm, where the gap after $i+1$ steps is a simple sum of local rounding errors, see (11). Consequently, like the three-term recurrence CG algorithm [33] analyzed in e.g. [20], the CG-CG recurrences may suffer from a strong amplification of local errors throughout the algorithm.

2.3 Propagation of rounding errors in pipelined CG

In preconditioned pipelined CG, Algorithm 3, additional recurrences are introduced for the auxiliary variables $w_i = Au_i$, $z_i = Aq_i$, $u_i = M^{-1}r_i$ and $q_i = M^{-1}s_i$, whereas $n_i = Am_i$ and $m_i = M^{-1}w_i$ are computed explicitly. In addition to avoiding communication, Alg. 3 hides the communication phase (lines 4-5) behind the SPMV and preconditioner application (lines 6-7). Replacing the recurrences by their finite precision equivalents, we have

$$\begin{aligned} x_{i+1} &= x_i + \alpha_i p_i + \delta_i^x, & p_i &= u_i + \beta_i p_{i-1} + \delta_i^p, \\ r_{i+1} &= r_i - \alpha_i s_i + \delta_i^r, & s_i &= w_i + \beta_i s_{i-1} + \delta_i^s, \\ w_{i+1} &= w_i - \alpha_i z_i + \delta_i^w, & z_i &= Am_i + \beta_i z_{i-1} + \delta_i^z, \\ u_{i+1} &= u_i - \alpha_i q_i + \delta_i^u, & q_i &= m_i + \beta_i q_{i-1} + \delta_i^q, \end{aligned} \tag{26}$$

where the local rounding errors are bounded by

$$\begin{aligned} \|\delta_i^x\| &\leq (\|x_i\| + 2\,|\alpha_i|\,\|p_i\|)\,\epsilon, \\ \|\delta_i^p\| &\leq (\|u_i\| + 2\,|\beta_i|\,\|p_{i-1}\|)\,\epsilon, \\ \|\delta_i^r\| &\leq (\|r_i\| + 2\,|\alpha_i|\,\|s_i\|)\,\epsilon, \\ \|\delta_i^s\| &\leq (\|w_i\| + 2\,|\beta_i|\,\|s_{i-1}\|)\,\epsilon, \\ \|\delta_i^w\| &\leq (\|w_i\| + 2\,|\alpha_i|\,\|z_i\|)\,\epsilon, \\ \|\delta_i^z\| &\leq \left((m\sqrt{n} + \tilde{m}\sqrt{n} + 1)\,\|A\|\,\|M^{-1}\|\,\|w_i\| + 2\,|\beta_i|\,\|z_{i-1}\|\right)\epsilon, \\ \|\delta_i^u\| &\leq (\|u_i\| + 2\,|\alpha_i|\,\|q_i\|)\,\epsilon, \\ \|\delta_i^q\| &\leq \left((\tilde{m}\sqrt{n} + 1)\,\|M^{-1}\|\,\|w_i\| + 2\,|\beta_i|\,\|q_{i-1}\|\right)\epsilon. \end{aligned} \tag{27}$$

The gap $f_i = (b - Ax_i) - r_i$ can then be calculated similarly to (18). The initial gap f_0 satisfies (17), and in iteration i we have

$$\begin{aligned} f_{i+1} &= (b - Ax_{i+1}) - r_{i+1} \\ &= b - A(x_i + \alpha_i p_i + \delta_i^x) - (r_i - \alpha_i s_i + \delta_i^r) \\ &= f_i - \alpha_i g_i - A\delta_i^x - \delta_i^r. \end{aligned} \tag{28}$$

The residual gap is again coupled to the error $g_i = Ap_i - s_i$, which can be written as

$$\begin{aligned} g_i &= Ap_i - s_i \\ &= A(u_i + \beta_i p_{i-1} + \delta_i^p) - (w_i + \beta_i s_{i-1} + \delta_i^s) \\ &= h_i + \beta_i g_{i-1} + A\delta_i^p - \delta_i^s, \end{aligned} \tag{29}$$

Algorithm 3 Preconditioned pipelined CG

procedure PREC-P-CG(A, M^{-1}, b, x_0)
  $r_0 := b - Ax_0$; $u_0 := M^{-1}r_0$; $w_0 := Au_0$
  for $i = 0, \ldots$ **do**
    $\gamma_i := (r_i, u_i)$
    $\delta := (w_i, u_i)$
    $m_i := M^{-1}w_i$
    $n_i := Am_i$
    if $i > 0$ **then**
      $\beta_i := \gamma_i/\gamma_{i-1}$; $\alpha_i := (\delta/\gamma_i - \beta_i/\alpha_{i-1})^{-1}$
    else
      $\beta_i := 0; \alpha_i = \gamma_i/\delta$
    end if
    $z_i := n_i + \beta_i z_{i-1}$
    $q_i := m_i + \beta_i q_{i-1}$
    $s_i := w_i + \beta_i s_{i-1}$
    $p_i := u_i + \beta_i p_{i-1}$
    $x_{i+1} := x_i + \alpha_i p_i$
    $r_{i+1} := r_i - \alpha_i s_i$
    $u_{i+1} := u_i - \alpha_i q_i$
    $w_{i+1} := w_i - \alpha_i z_i$
  end for
end procedure

where g_0 satisfies (19) and we define $h_i = Au_i - w_i$. Instead of being computed explicitly, the auxiliary variable $w_i = Au_i$ is also computed recursively in pipelined CG, leading to an additional coupling of the residual gap f_i to the rounding error h_i. For $i = 0$, it holds that the norm of the gap on w_i is bounded by

$$\|h_0\| \leq m\sqrt{n}\,\|A\|\,\|u_0\|\,\epsilon. \tag{30}$$

Substituting the recurrences (26), we find that the gap on w_{i+1} in iteration i is

$$\begin{aligned} h_{i+1} &= Au_{i+1} - w_{i+1} \\ &= A(u_i - \alpha_i q_i + \delta_i^u) - (w_i - \alpha_i z_i + \delta_i^w) \\ &= h_i - \alpha_i j_i + A\delta_i^u - \delta_i^w, \end{aligned} \tag{31}$$

which relates the residual error to the error $j_i = Aq_i - z_i$. The latter error is due to the recursive computation of the auxiliary variable $z_i = Aq_i$. For $i = 0$, we can estimate the norm of j_i as the roundoff, i.e.

$$\|j_0\| \leq m\sqrt{n}\,\|A\|\,\|q_0\|\,\epsilon. \tag{32}$$

Using again the recursive definitions (26), we obtain

$$\begin{aligned} j_i &= Aq_i - z_i \\ &= A(m_i + \beta_i q_{i-1} + \delta_i^q) - (n_i + \beta_i z_{i-1} + \delta_i^z) \\ &= \beta_i j_{i-1} + A\delta_i^q - \delta_i^z, \end{aligned} \tag{33}$$

where the final equation holds since $n_i = Am_i$ is computed explicitly in Algorithm 3. Hence, for

pipelined CG, the residual gap is given by the system of coupled equations

$$\begin{bmatrix} f_{i+1} \\ g_i \\ h_{i+1} \\ j_i \end{bmatrix} = \begin{bmatrix} 1 & -\alpha_i\beta_i & -\alpha_i & 0 \\ 0 & \beta_i & 1 & 0 \\ 0 & 0 & 1 & -\alpha_i\beta_i \\ 0 & 0 & 0 & \beta_i \end{bmatrix} \begin{bmatrix} f_i \\ g_{i-1} \\ h_i \\ j_{i-1} \end{bmatrix} + \begin{bmatrix} -A\delta_i^x - \delta_i^r - \alpha_i \left(A\delta_i^p - \delta_i^s\right) \\ A\delta_i^p - \delta_i^s \\ A\delta_i^u - \delta_i^w - \alpha_i \left(A\delta_i^q - \delta_i^z\right) \\ A\delta_i^q - \delta_i^z \end{bmatrix}. \tag{34}$$

Taking the norms of both sides, we arrive at the following coupled system of upper bounds for the gaps in pipelined CG:

$$\begin{bmatrix} \|f_{i+1}\| \\ \|g_i\| \\ \|h_{i+1}\| \\ \|j_i\| \end{bmatrix} \leq \begin{bmatrix} 1 & |\alpha_i\beta_i| & |\alpha_i| & 0 \\ 0 & |\beta_i| & 1 & 0 \\ 0 & 0 & 1 & |\alpha_i\beta_i| \\ 0 & 0 & 0 & |\beta_i| \end{bmatrix} \begin{bmatrix} \|f_i\| \\ \|g_{i-1}\| \\ \|h_i\| \\ \|j_{i-1}\| \end{bmatrix} + \begin{bmatrix} \|A\delta_i^x + \delta_i^r\| + |\alpha_i| \|A\delta_i^p - \delta_i^s\| \\ \|A\delta_i^p - \delta_i^s\| \\ \|A\delta_i^u - \delta_i^w\| + |\alpha_i| \|A\delta_i^q - \delta_i^z\| \\ \|A\delta_i^q - \delta_i^z\| \end{bmatrix}. \tag{35}$$

The per-iteration additions on the right-hand side in (35) can be bounded further using the local error bounds (27). We hence obtain the computable bounds

$$\begin{aligned} \|A\delta_i^x + \delta_i^r\| &\leq \|A\| \|\delta_i^x\| + \|\delta_i^r\| \\ &\leq \left(\|A\| \|x_i\| + 2 |\alpha_i| \|A\| \|p_i\| + \|r_i\| + 2 |\alpha_i| \|s_i\|\right) \epsilon \\ &:= e_i^f \epsilon, \end{aligned} \tag{36}$$

$$\begin{aligned} \|A\delta_i^p - \delta_i^s\| &\leq \|A\| \|\delta_i^p\| + \|\delta_i^s\| \\ &\leq \left(\|A\| \|u_i\| + 2 |\beta_i| \|A\| \|p_{i-1}\| + \|w_i\| + 2 |\beta_i| \|s_{i-1}\|\right) \epsilon \\ &:= e_i^g \epsilon, \end{aligned} \tag{37}$$

$$\begin{aligned} \|A\delta_i^u - \delta_i^w\| &\leq \|A\| \|\delta_i^u\| + \|\delta_i^w\| \\ &\leq \left(\|A\| \|u_i\| + 2 |\alpha_i| \|A\| \|q_i\| + \|w_i\| + 2 |\alpha_i| \|z_i\|\right) \epsilon \\ &:= e_i^h \epsilon, \end{aligned} \tag{38}$$

and

$$\begin{aligned} \|A\delta_i^q - \delta_i^z\| &\leq \|A\| \|\delta_i^q\| + \|\delta_i^z\| \\ &\leq \big((\tilde{m}\sqrt{n} + 1) \|A\| \|M^{-1}\| \|w_i\| + 2 |\beta_i| \|A\| \|q_{i-1}\| \\ &\quad + (m\sqrt{n} + \tilde{m}\sqrt{n} + 1) \|A\| \|M^{-1}\| \|w_i\| + 2 |\beta_i| \|z_{i-1}\|\big) \epsilon \\ &= \big(2 |\beta_i| \|A\| \|q_{i-1}\| + ((m + 2\tilde{m})\sqrt{n} + 2) \|A\| \|M^{-1}\| \|w_i\| \\ &\quad + 2 |\beta_i| \|z_{i-1}\|\big) \epsilon \\ &:= e_i^j \epsilon. \end{aligned} \tag{39}$$

This yields the following system of upper bounds on the norm of the gap between the true and recursive residual:

$$\begin{bmatrix} \|f_{i+1}\| \\ \|g_i\| \\ \|h_{i+1}\| \\ \|j_i\| \end{bmatrix} \leq \begin{bmatrix} 1 & |\alpha_i\beta_i| & |\alpha_i| & 0 \\ 0 & |\beta_i| & 1 & 0 \\ 0 & 0 & 1 & |\alpha_i\beta_i| \\ 0 & 0 & 0 & |\beta_i| \end{bmatrix} \begin{bmatrix} \|f_i\| \\ \|g_{i-1}\| \\ \|h_i\| \\ \|j_{i-1}\| \end{bmatrix} + \begin{bmatrix} e_i^f \epsilon + |\alpha_i| e_i^g \epsilon \\ e_i^g \epsilon \\ e_i^h \epsilon + |\alpha_i| e_i^j \epsilon \\ e_i^j \epsilon \end{bmatrix}, \tag{40}$$

where e_i^f, e_i^g, e_i^h and e_i^j are defined in (36)-(39).

Note that the auxiliary variables $u_i = M^{-1}r_i$ and $q_i = M^{-1}s_i$, which represent the preconditioned versions of the residual r_i and the auxiliary variable s_i respectively, are also computed recursively in pipelined CG. Hence, we can write down an analogous derivation for the gap between the explicit and recursive preconditioned residual, that is, $k_i = M^{-1}r_i - u_i$. For $i = 0$ we have

$$\|k_0\| \leq \tilde{m}\sqrt{n}\,\|M^{-1}\|\,\|r_0\|\,\epsilon, \tag{41}$$

and in iteration i we find

$$\begin{aligned} k_{i+1} &= M^{-1}r_{i+1} - u_{i+1} \\ &= M^{-1}(r_i - \alpha_i s_i + \delta_i^r) - (u_i - \alpha_i q_i + \delta_i^u) \\ &= k_i - \alpha_i \ell_i + M^{-1}\delta_i^r - \delta_i^u, \end{aligned} \tag{42}$$

where we define $\ell_i = M^{-1}s_i - q_i$. Finally, for the gap between the explicit and recursive q_i, we have for $i = 0$ that

$$\|\ell_0\| \leq \tilde{m}\sqrt{n}\,\|M^{-1}\|\,\|s_0\|\,\epsilon. \tag{43}$$

By once again inserting the recurrences from (26), we find that ℓ_i in iteration i is

$$\begin{aligned} \ell_i &= M^{-1}s_i - q_i \\ &= M^{-1}(w_i + \beta_i s_{i-1} + \delta_i^s) - (m_i + \beta_i q_{i-1} + \delta_i^q) \\ &= \beta_i \ell_{i-1} + M^{-1}\delta_i^s - \delta_i^q. \end{aligned} \tag{44}$$

The last equation in (44) holds since $m_i = M^{-1}w_i$ is computed explicitly in Algorithm 3. This leads to a separate system of coupled recurrences for the gap k_{i+1}:

$$\begin{bmatrix} k_{i+1} \\ \ell_i \end{bmatrix} = \begin{bmatrix} 1 & -\alpha_i\beta_i \\ 0 & \beta_i \end{bmatrix} \begin{bmatrix} k_i \\ \ell_{i-1} \end{bmatrix} + \begin{bmatrix} M^{-1}\delta_i^r - \delta_i^u - \alpha_i\left(M^{-1}\delta_i^s - \delta_i^q\right) \\ M^{-1}\delta_i^s - \delta_i^q \end{bmatrix}. \tag{45}$$

However, since the gap k_{i+1} is uncoupled from the residual gap f_{i+1}, the coupled recurrences (42)-(44) are not be taken into account when establishing bounds for the norm of the residual gap in (40).

3 Estimating the gap between true and recursive residuals

The analysis in the previous sections shows that, due to the addition of additional recurrence relations in the pipelined CG algorithm, the propagation of local rounding errors through Algorithm 3 may be dramatic. We thus aim to introduce countermeasures to this unstable behavior in the form of an automated residual replacement strategy. In order to formulate such an automated strategy, a sufficiently good and practically computable estimate for the residual gap should be available.

3.1 A practical estimate for the residual gap

In the previous sections we have derived upper bounds for the norm of the residual gap for several variants of the CG algorithm. Although insightful from an analytical perspective, these bounds are typically not sharp. Indeed, the bounds on the norms of the local rounding errors (27) may largely overestimate the actual error norms, see the discussion in [19], p. 541. Hence, the equality in e.g. (40) far from holds, and the right-hand side bounds provide poor estimates for the actual residual gap. To obtain a more realistic estimate for the residual gap, we turn

to statistical analysis of the rounding errors, see [37]. A well-known rule of thumb [23] is that a realistic error estimate can be obtained by replacing the dimension-dependent constants in a rounding error bound by their square root; thus if the bound is $f(n)\epsilon$, the error is approximately of order $\sqrt{f(n)}\epsilon$. Hence, instead of using the upper bounds (36)-(39), we use the following approximations for the local error norms:

$$\begin{aligned} \|A\delta_i^x + \delta_i^r\| &\approx \sqrt{e_i^f}\,\epsilon, & \|A\delta_i^p - \delta_i^s\| &\approx \sqrt{e_i^g}\,\epsilon, \\ \|A\delta_i^u + \delta_i^w\| &\approx \sqrt{e_i^h}\,\epsilon, & \|A\delta_i^q - \delta_i^z\| &\approx \sqrt{e_i^j}\,\epsilon. \end{aligned} \tag{46}$$

Note that for the norms of the initial gaps in (17), (19), (30) and (32), a similar square root rescaling of the respective dimension-dependent factors has to be applied. We hence assume that the norm of the residual gap in iteration i of the pipelined CG algorithm can be estimated as follows:

$$\begin{bmatrix} \|f_{i+1}\| \\ \|g_i\| \\ \|h_{i+1}\| \\ \|j_i\| \end{bmatrix} \approx \begin{bmatrix} 1 & |\alpha_i\beta_i| & |\alpha_i| & 0 \\ 0 & |\beta_i| & 1 & 0 \\ 0 & 0 & 1 & |\alpha_i\beta_i| \\ 0 & 0 & 0 & |\beta_i| \end{bmatrix} \begin{bmatrix} \|f_i\| \\ \|g_{i-1}\| \\ \|h_i\| \\ \|j_{i-1}\| \end{bmatrix} + \begin{bmatrix} \sqrt{e_i^f}\,\epsilon + |\alpha_i|\sqrt{e_i^g}\,\epsilon \\ \sqrt{e_i^g}\,\epsilon \\ \sqrt{e_i^h}\,\epsilon + |\alpha_i|\sqrt{e_i^j}\,\epsilon \\ \sqrt{e_i^j}\,\epsilon \end{bmatrix}. \tag{47}$$

This approximation tends to yield a good estimate for the actual residual gap, as illustrated by the numerical experiments in the next section.

A second practical remark concerns the computation of the matrix and preconditioner norms, $\|A\|$ and $\|M^{-1}\|$, in the estimate (47) for the gap between the true and recursive residual. The use of the matrix 2-norm is often prohibited in practice, since it is computationally expensive for large scale systems. However, using the norm inequality $\|A\|_2 \leq \|A\|_\infty$, the matrix 2-norms in the estimate can be replaced by their respective maximum norms. This slightly worsens the estimate, but provides practically computable quantities for e_i^f, e_i^g, e_i^h and e_i^j. Alternatively, in the context of matrix-free computations, randomized probabilistic techniques for matrix norm computation may be used, see for example [21].

A related issue concerns the computation of the norm of the preconditioner. The operator M^{-1} is often not available in matrix form. This is the case when preconditioning the system with e.g. an Incomplete Cholesky factorization (ICC) type scheme, or any (stencil-based) scheme where M^{-1} is not explicitly formed. For these commonly used preconditioning methods the norm $\|M^{-1}\|$ is unavailable. Explicit use of the preconditioner norm in the estimate (47) can be avoided by reformulating the local rounding error bounds (27) in function of the preconditioned variable $m_i = M^{-1}w_i$, that is:

$$\begin{aligned} \|\delta_i^z\| &\leq \left((m\sqrt{n}+1)\,\|A\|\,\|m_i\| + 2\,|\beta_i|\,\|z_{i-1}\|\right)\epsilon, \\ \|\delta_i^q\| &\leq \left(\|m_i\| + 2\,|\beta_i|\,\|q_{i-1}\|\right)\epsilon. \end{aligned} \tag{48}$$

These bounds do not explicitly take the rounding error of the multiplication $M^{-1}w_i$ into account, but rather implicitly bound the local rounding error using $\|m_i\|$. With these local rounding error bounds, e_i^j can now be defined analogous to (39) as

$$\begin{aligned} \|A\delta_i^q - \delta_i^z\| &\leq \|A\|\,\|\delta_i^q\| + \|\delta_i^z\| \\ &\leq \big(\|A\|\,\|m_i\| + 2\,|\beta_i|\,\|A\|\,\|q_{i-1}\| \\ &\quad + (m\sqrt{n}+1)\,\|A\|\,\|m_i\| + 2\,|\beta_i|\,\|z_{i-1}\|\big)\,\epsilon \\ &= \left((m\sqrt{n}+2)\,\|A\|\,\|m_i\| + 2\,|\beta_i|\,\|A\|\,\|q_{i-1}\| + 2\,|\beta_i|\,\|z_{i-1}\|\right)\epsilon \\ &:= e_i^j\epsilon. \end{aligned} \tag{49}$$

With e_i^j defined as in (49), the estimate (47) can be used to predict the residual gap in pipelined CG when the preconditioning matrix M^{-1} is not formed explicitly.

Finally, it should be pointed out that the error factors e_i^f, e_i^g, e_i^h and e_i^j defined in (36)-(39) and (49) contain several norms that are by default not computed in Algorithm 3. Their computation causes no additional communication overhead, since they can be combined with the existing global communication phase on lines 4-5, and virtually no computational overhead due to the low flop count per dot-product. However, the norms of some auxiliary variables, notably p_i, s_i, q_i, z_i and m_i, are unavailable in step i, since these vectors are not defined until *after* the global reduction phase. Hence, their norms can be computed at the earliest in the global reduction phase of iteration $i+1$. Consequently, in practical implementations the estimated norm $\|f_{i+1}\|$, defined by (47), for the gap between the true residual $b - Ax_{i+1}$ and the recursive residual r_{i+1} from iteration i can only be computed in iteration $i+1$. This means that when including the estimates for the residual gap into the pipelined CG algorithm, a delay of one iteration on the estimates is unavoidable.

3.2 Pipelined CG with automated residual replacement

In this section we propose an automated residual replacement strategy for pipelined CG, based on the estimates for the gap between the true and recursive residual (47). This countermeasure reduces the propagation of rounding errors, and hence improves the possibly dramatically reduced maximal attainable accuracy of the pipelined method. The idea of performing manual residual replacements to increase the attainable accuracy of pipelined CG was already suggested in the original paper [17]. However, the proper choice of the replacement iterations requires specific knowledge on the convergence of CG for the particular system.

The cause of the early stagnation of the true residual is traced back to the propagation of local rounding errors by the analysis in the previous section. Note, however, that the residual may additionally show irregular jumps beyond the stagnation point. The origin of this behavior can also be found in the rounding error analysis of the residual gap, and its effect on the recurrence for the scalars α_i and β_i. We briefly assume $M = I$ for notational convenience. Algorithm 3 uses $\gamma_i := (r_i, r_i)$ to calculate an update for α_i and β_i. Here β_i is traditionally defined as $\gamma_i/\gamma_{i-1} = (r_i, r_i)/(r_{i-1}, r_{i-1})$. Furthermore, $\alpha_i := \gamma_i/(p_i, Ap_i)$, such that $(p_i, Ap_i) = \gamma_i/\alpha_i$. This yields

$$\delta := (r_i, Ar_i) = ((p_i - \beta_i p_{i-1}), A(p_i - \beta_i p_{i-1})) = (p_i, Ap_i) + \beta_i^2 (p_{i-1}, Ap_{i-1}), \tag{50}$$

where we used that the search directions p_i are A-orthogonal to eliminate (p_i, Ap_{i-1}). We obtain

$$\delta = \frac{\gamma_i}{\alpha_i} + \beta_i^2 \frac{\gamma_{i-1}}{\alpha_{i-1}} = \frac{\gamma_i}{\alpha_i} + \beta_i \frac{\gamma_i}{\alpha_{i-1}}. \tag{51}$$

Reorganizing the above directly leads to the relation $\alpha_i = (\delta/\gamma_i - \beta_i/\alpha_{i-1})^{-1}$ which is used in Algorithm 3 on line 9. When the algorithm is close to convergence, rounding errors dominate the update for the residual, see (34). This corrupts the (orthogonality) relations that typically hold between subsequent residuals, leading to inaccuracies in α_i and β_i. As a result, incorrect residuals and even divergence from the solution may arise around the stagnation point, resulting in residual 'peaks'. However, a movement away from the solution results in a growth of (r_i, r_i). Consequently, after a small number of iterations the scalars α_i and β_i can again be accurately calculated, leading to renewed convergence and a reduction in the residuals. When the residual rounding errors again become dominant, the cycle restarts and a new residual peak is formed.

We propose an automated residual replacement strategy to improve the maximal attainable accuracy of pipelined CG and alleviate the residual irregularities after the stagnation point

described above. We follow the basic idea of residual replacement in Krylov subspace methods as discussed by Van der Vorst et al. [36] and Sleijpen et al. [31, 32]. In specific iterations of the algorithm, the vectors r_{i+1}, w_{i+1}, u_{i+1}, s_i, z_i and q_i which are computed recursively in iteration i, are instead computed explicitly as

$$\begin{aligned} r_{i+1} &= b - Ax_{i+1}, & s_i &= Ap_i, \\ w_{i+1} &= Au_{i+1}, & z_i &= Aq_i, \\ u_{i+1} &= M^{-1}r_{i+1} & q_i &= M^{-1}s_i. \end{aligned} \tag{52}$$

Note how the current iterate x_{i+1} and the search direction p_i are evidently not replaced, since no explicit formulae for these vectors are available.

Two important caveats arise when incorporating a residual replacement in an iterative method. First, one could inquire if such a drastic replacement strategy does not destroy the established Krylov convergence. A second, related question concerns the use of a criterion for the iteration in which replacements should take place. Since each residual replacement step comes at an additional cost of computing the SPMVs in (52), an accurate criterion to determine the need for residual replacement that does not overestimate the total number of replacements is essential.

We briefly recapitulate the primary results from [35] and [36] below. The recurrences for r_{i+1} and p_i in the (unpreconditioned) Algorithm 1 are

$$\begin{aligned} r_{i+1} &= r_i - \alpha_i A p_i + \delta_i^r, \\ p_{i+1} &= r_{i+1} + \beta_{i+1} p_i + \delta_i^p, \end{aligned} \tag{53}$$

where δ_i^r and δ_i^p are bounded by (8) . Combining the above recursions yields the perturbed Lanczos relation

$$AZ_i = Z_i T_i - \frac{\|r_0 - \alpha_0 A p_0\|}{\alpha_i \|r_1\| \|r_i\|} r_{i+1} e_i^T + F_i \quad \text{with} \quad Z_i = \left[\frac{r_1}{\|r_1\|}, \ldots, \frac{r_i}{\|r_i\|} \right], \tag{54}$$

see [36], where T_i is a tridiagonal matrix and F_i is a perturbation caused by the local rounding errors. We assume that Z_i is full rank and that no breakdowns have occurred. A key result from [35] states that if r_i satisfies the perturbed Lanczos relation (54), then

$$\|r_{i+1}\| \leq C_i \min_{p \in \mathcal{P}_i, p(0)=1} \|p(A - F_i Z_i^+)\, r_1\|, \tag{55}$$

where $C_i > 0$ is an iteration-dependent constant. This result implies that even if the perturbation F_i is significantly larger than ϵ, which is the case after residual replacement, the norm of the residual is not significantly affected and convergence remains stable. Based on the bound (55), Van der Vorst et al. propose in [36] to explicitly update the residuals and other vectors only when the residual norm is sufficiently large compared to the norm of the rounding error. Convergence is then expected to resume in a similar fashion. Performing replacements when $\|r_i\|$ is small is not recommended, since this could negatively affect the CG convergence.

A replacement in step i eliminates the residual gap f_{i+1}. However, it should be carried out before $\|f_i\|$ becomes too large relative to $\|r_i\|$. In analogy to [36], we use a threshold τ, typically chosen as $\tau = \sqrt{\epsilon}$, and perform a residual replacement in step i of Algorithm 3 if

$$\|f_{i-1}\| \leq \tau \|r_{i-1}\| \quad \text{and} \quad \|f_i\| > \tau \|r_i\|. \tag{56}$$

This criterion ensures that convergence is maintained when a residual replacement takes places, since replacements are only allowed when $\|r_i\|$ is sufficiently large with respect to $\|f_i\|$. Furthermore, no excess replacement steps are performed, thus keeping the total cost of the algorithm as

Matrix	n	CG		CG-CG		p-CG		p-CG-rr		
		iter	res	iter	res	iter	res	iter	res	rr
lapl50	2,500	118	2.4e-13	119	2.5e-13	111	1.6e-11	117	2.1e-13	2
lapl100	10,000	230	1.3e-12	234	1.3e-12	213	2.6e-10	243	1.2e-12	3
lapl200	40,000	453	7.2e-12	453	7.1e-12	407	4.5e-09	472	6.9e-12	3
lapl400	160,000	886	3.9e-11	886	3.9e-11	773	6.9e-08	1024	8.8e-11	6
lapl800	640,000	1727	2.1e-10	1717	2.3e-10	1375	1.1e-06	1783	5.9e-10	6

Table 1: *Model problem 2D Laplacian operators of various sizes. A linear system with right-hand side* $b = A\hat{x}$ *where* $\hat{x}_j = 1/\sqrt{n}$ *is solved for each of these matrices with the four presented algorithms. The initial guess is all-zero* $x_0 = 0$*. The number of iterations required to reach maximal attainable accuracy is given, along with the corresponding true residual norm* $\|b - Ax_i\|_2$*. For the p-CG-rr method the number of replacement steps* rr *is indicated.*

low as possible. The residual gap model (47) allows for the practical implementation of the replacement criterion (56) in Algorithm 3. Note that criterion (56) does not compare the estimate $\|f_{i+1}\|$ to the current residual norm $\|r_{i+1}\|$ computed in iteration i, since the norms of both these vectors are not yet available, see the discussion near the end of Section 3.1. This implies that the primary cost of performing the automated replacement strategy is not a communication or computation overhead, but the storage of these auxiliary variables between subsequent iterations. The resulting algorithm is called pipelined CG with automated residual replacement (p-CG-rr).

4 Numerical results

Numerical results on a wide range of matrices from applications are presented to compare the convergence of the different CG methods and show the improved attainable accuracy using the automated residual replacement strategy. The convergence results in Sections 4.1 and 4.2 are based on a Matlab implementation of the different CG algorithms and error estimates. Parallel performance measurements in Section 4.3 result from a PETSc implementation of pipelined CG with automated residual replacements on a distributed memory machine using the message passing paradigm.

4.1 Poisson model problem

The methods presented above are tested on a two-dimensional Laplacian PDE model with homogeneous Dirichlet boundary conditions, discretized using second order finite differences on a uniform $n = n_x \times n_y$ point discretization of the unit square. Table 1 shows convergence results for solving the discrete Poisson system for $n_x = n_y = 50, 100, 200, 400$ and 800, with condition numbers ranging from 1.5e+3 to 1.8e+5. A linear system with exact solution $\hat{x}_j = 1/\sqrt{n}$ (such that $\|\hat{x}\| = 1$) and right-hand side $b = A\hat{x}$ is solved for each of these discretization matrices. The initial guess is $x_0 = 0$. The iteration is stopped when maximal attainable accuracy, i.e. $\min_i \|b - Ax_i\|_2$, is reached. This implies that a different stopping tolerance is used for each matrix. No preconditioner is used for the Laplace problems. The table lists the required number of iterations *iter* and the final true residual norm *res* for the CG, Chronopoulos/Gear CG, p-CG and p-CG-rr methods. Pipelined CG stagnates at a significantly larger residual compared to CG and CG-CG due to the propagation of rounding errors from the auxiliary recursions, see (34). Note that for larger systems the loss of accuracy is dramatically more pronounced.

Figure 1 shows the A-norm of the error in function of iterations for the `lapl100` and `lapl400` problems from Table 1. The CG method minimizes this quantity over the respective Krylov subspace in each iteration, which (in exact arithmetic) results in a monotonically decreasing error norm. For the pipelined CG method, the error norm behaves similar to the CG method up

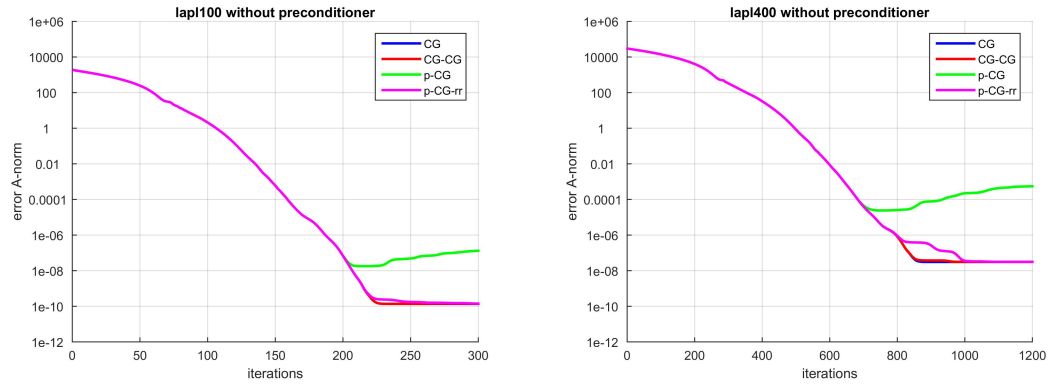


Figure 1: *Error history for the different CG methods applied to the* `lapl100` *(left) and* `lapl400` *(right) matrices, see Table 1. Error A-norm* $\|\hat{x}-x_i\|_A$ *in function of iterations for CG (blue), Chronopoulos/Gear CG (red), pipelined CG (green) and p-CG-rr (magenta).*

to its stagnation point. Beyond this point the method becomes unstable due to rounding error propagation and the error norm is no longer guaranteed to decrease. Periodic replacement of the residual and auxiliary variables resolves the instability, as illustrated by the monotonically decreasing p-CG-rr errors. However, a slight delay of convergence [34] is observed for the p-CG-rr method compared to classical CG, see also Table 1. We discuss the effect of rounding errors on orthogonality and the resulting delay of convergence near the end of Section 4.2. Since the error is in general unavailable, the remaining experiments focus on the norm of the residual instead.

Figure 2 illustrates the residual convergence history and the corresponding gaps between the explicit variables and their recursive counterparts in the different CG algorithms for the `lapl50` matrix. Convergence of CG and Chronopoulos/Gear CG is nearly indistinguishable. For CG and CG-CG, the norm of the true residual at stagnation is 2.4e-13 and 2.5e-13 respectively, see Table 1. The pipelined CG method suffers from the propagation of rounding errors, leading to early stagnation of the residual norm at 1.6e-11. The residual replacement strategy reduces the effects of rounding errors, leading to a residual norm of 2.1e-13, comparable to standard CG.

4.2 Problems from Matrix Market

Numerical results on a wide range of linear systems are presented to show the effectiveness of pipelined CG with automated residual replacements. Table 2 lists all real, non-diagonal and symmetric positive definite matrices from Matrix Market[1], with their respective condition number κ, number of rows n and total number of nonzero elements $\#nnz$. We solve a linear system with exact solution $\hat{x}_j = 1/\sqrt{n}$ and right-hand side $b = A\hat{x}$ for each of these matrices with the four presented methods, using an all-zero initial guess $x_0 = 0$. Jacobi diagonal preconditioning (JAC) and Incomplete Cholesky Factorization (ICC) are included to reduce the number of Krylov iterations if possible. For the preconditioners designated by *ICC an compensated Incomplete Cholesky factorization is performed, where a real non-negative scalar η is used as a global diagonal shift in forming the Cholesky factor. For the `nos1` and `nos2` matrices the shift is $\eta = 0.5$, whereas for all other *ICC preconditioners we used $\eta = 0.1$.

[1] `http://math.nist.gov/MatrixMarket/`

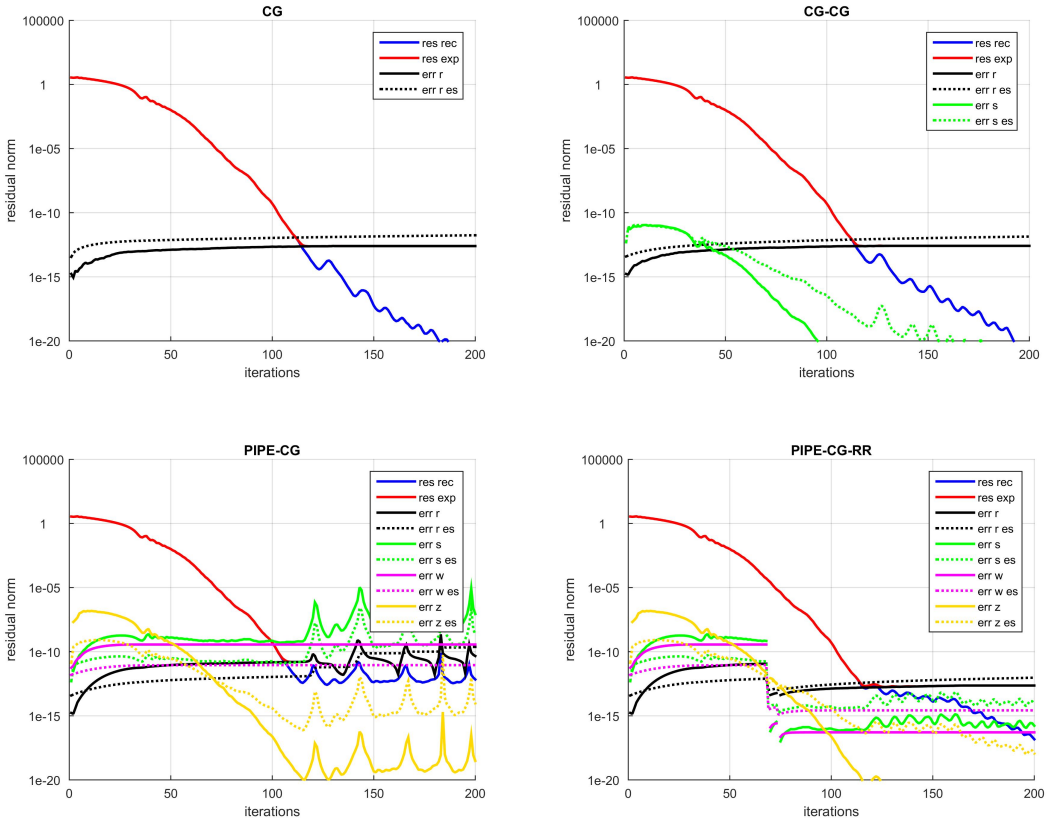


Figure 2: *Convergence history for the different CG methods applied to the* `lapl50` *matrix, see Table 1. Blue: recursive residual norm* $\|r_i\|_2$. *Red: true residual norm* $\|b - Ax_i\|_2$. *Solid lines (black/green/magenta/yellow): gap norms of residual and auxiliary vector gaps (computed explicitly). Dashed lines: gap estimates* (47).

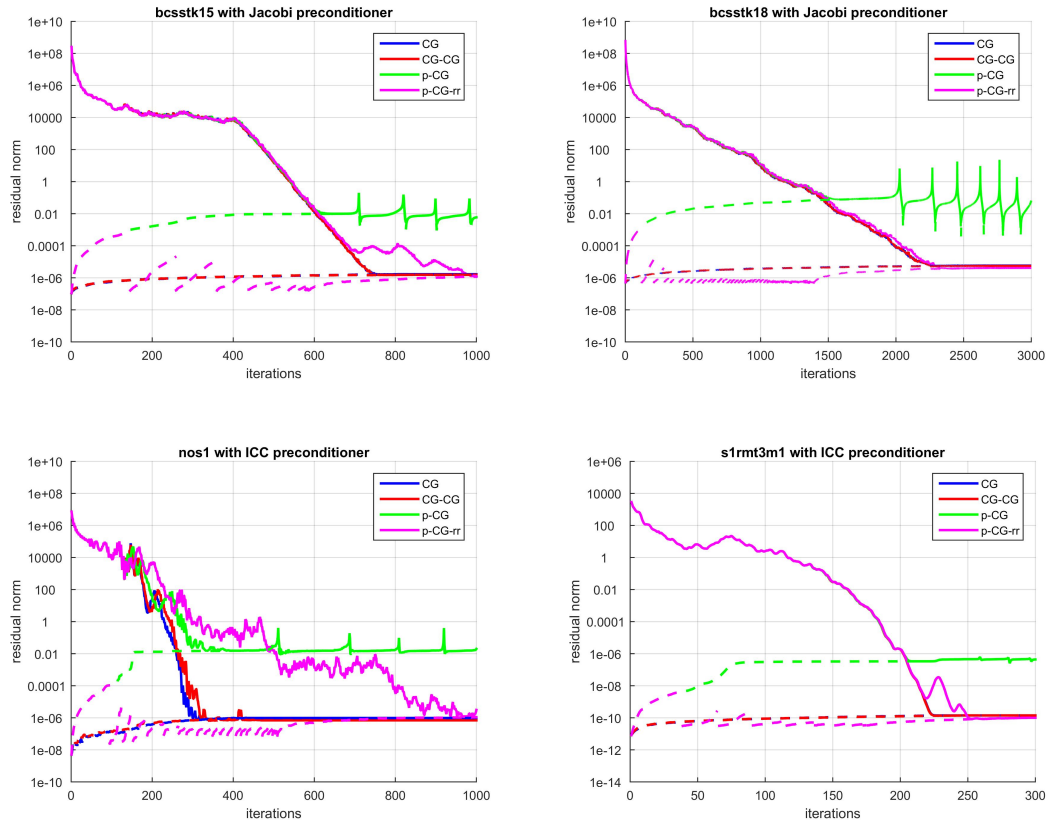


Figure 3: *Convergence history for the different CG methods applied to four symmetric positive definite test matrices from Table 2). Solid lines: true residual norm* $\|b-Ax_i\|_2$*; dashed lines: residual gap* $\|f_{i+1}\|_2$*. Convergence of CG (blue) and Chronopoulos/Gear CG (red) is largely comparable. The pipelined CG method (green) suffers from rounding error propagation. Automated residual replacement (magenta) reduces the rounding errors, leading to an accuracy that is comparable to standard CG.*

Matrix	Prec	$\kappa(A)$	n	$\#nnz$	$\|\|r_0\|\|$	CG		CG-CG		p-CG		p-CG-rr		
						iter	res	iter	res	iter	res	iter	res	rr
bcsstk14	JAC	1.3e+10	1806	63,454	2.1e+09	650	1.6e-06	658	1.5e-06	506	1.1e-02	658	1.1e-06	9
bcsstk15	JAC	8.0e+09	3948	117,816	4.3e+08	772	1.6e-06	785	1.5e-06	646	9.9e-03	974	1.7e-06	10
bcsstk16	JAC	65	4884	290,378	1.5e+08	298	5.3e-07	300	6.0e-07	261	1.3e-03	301	3.2e-07	4
bcsstk17	JAC	65	10,974	428,650	9.0e+07	3547	9.1e-07	3428	1.5e-06	2913	2.5e-01	4508	1.1e-06	54
bcsstk18	JAC	65	11,948	149,090	2.6e+09	2299	5.6e-06	2294	5.4e-06	1590	7.6e-02	2400	3.8e-06	50
bcsstk27	JAC	7.7e+04	1224	56,126	1.1e+05	345	3.5e-10	345	4.4e-10	295	8.8e-07	342	3.0e-10	6
gr_30_30	-	3.8e+02	900	7744	1.1e+00	56	3.0e-15	55	3.4e-15	52	2.2e-13	61	3.3e-15	2
nos1	*ICC	2.5e+07	237	1017	5.7e+07	301	7.5e-07	338	6.6e-07	337	1.5e-02	968	1.1e-06	21
nos2	*ICC	6.3e+09	957	4137	1.8e+09	3180	1.5e-04	3292	2.0e-04	2656	2.2e+02	4429	4.9e-02	113
nos3	ICC	7.3e+04	960	15,844	1.0e+01	64	1.0e-13	63	1.1e-13	59	1.0e-11	61	2.5e-13	3
nos4	ICC	2.7e+03	100	594	5.2e-02	31	9.7e-17	31	9.7e-17	29	2.1e-15	33	6.5e-17	2
nos5	ICC	2.9e+04	468	5172	2.8e+05	63	9.0e-11	64	9.4e-11	58	1.2e-08	65	6.3e-11	2
nos6	ICC	8.0e+06	675	3255	8.6e+04	34	4.4e-10	35	5.3e-10	31	4.7e-06	33	9.0e-10	2
nos7	ICC	4.1e+09	729	4617	8.6e+03	29	3.4e-10	31	2.4e-10	29	3.9e-10	29	2.6e-10	3
s1rmq4m1	ICC	1.8e+06	5489	262,411	1.5e+04	122	6.4e-11	122	6.9e-11	114	8.3e-08	135	5.5e-11	6
s1rmt3m1	ICC	2.5e+06	5489	217,651	1.5e+04	229	1.4e-10	228	1.3e-10	213	3.3e-07	240	2.5e-10	9
s2rmq4m1	*ICC	1.8e+08	5489	263,351	1.5e+03	370	1.0e-11	387	1.1e-11	333	4.1e-07	349	3.8e-10	25
s2rmt3m1	ICC	2.5e+08	5489	217,681	1.5e+03	285	1.3e-11	283	1.5e-11	250	1.1e-06	425	1.3e-11	17
s3dkq4m2	*ICC	1.9e+11	90,449	2,455,670	6.8e+01	-	1.3e-06	-	1.4e-06	-	1.9e-05	-	3.8e-06	199
s3dkt3m2	*ICC	3.6e+11	90,449	1,921,955	6.8e+01	-	2.0e-05	-	2.0e-05	-	2.4e-05	-	2.0e-05	252
s3rmq4m1	*ICC	1.8e+10	5489	262,943	1.5e+02	1651	2.3e-12	1789	2.4e-12	1716	3.9e-06	1602	7.9e-08	154
s3rmt3m1	*ICC	2.5e+10	5489	217,669	1.5e+02	2282	4.1e-12	2559	4.3e-12	2709	1.4e-05	3448	1.2e-07	149
s3rmt3m3	*ICC	2.4e+10	5357	207,123	1.3e+02	2862	4.3e-12	2798	4.4e-12	3378	2.7e-05	2556	9.2e-09	248

Table 2: *All real, non-diagonal and symmetric positive definite matrices from Matrix Market, listed with their respective condition number $\kappa(A)$, number of rows/columns n and total number of nonzeros $\#nnz$. A linear system with right-hand side $b = A\hat{x}$ where $\hat{x}_i = 1/\sqrt{n}$ is solved with each matrix with the four presented algorithms. The initial guess is all-zero $x_0 = 0$. Jacobi (JAC) and Incomplete Cholesky (ICC) preconditioners are included where needed. The number of iterations iter required to reach maximal attainable accuracy (stagnation point) and the corresponding true residual $res = \|b - Ax_i\|_2$ are shown. For the p-CG-rr method the number of replacement steps is indicated as rr.*

Table 2 lists the number of iterations *iter* required to reach maximal accuracy and the corresponding explicitly computed residual norm *res* for all methods. A '-' entry denotes failure to reach maximal accuracy within 5,000 iterations. The residual replacement strategy incorporated in p-CG-rr improves the attainable accuracy of the p-CG method, restoring accuracy to the order of the classical CG method.

Figure 3 illustrates the convergence history for several randomly selected matrices from Table 2. The top panels show the true residual (solid) and residual gap (dashed) for the `bcsstk15` and `bcsstk18` matrices with Jacobi preconditioning. The bottom panels show the convergence history for the `nos1` and `s1rmt3m1` matrices with ICC preconditioner. The propagation of rounding errors in the pipelined CG algorithm causes the residuals to level off sooner compared to standard and Chronopoulos/Gear CG. Based on the estimated residual gap (47), the residual replacement strategy explicitly computes the residual in the iterations where the criterion (56) is satisfied, leading to a more accurate final solution.

Note that the behavior of the p-CG and p-CG-rr residuals near the stagnation point differs slightly from the standard CG residuals, which is an artifact from the irregularities in the scalar coefficients α_i and β_i near stagnation, see Section 3.2. Furthermore, we point out that for the `nos1` matrix, see Fig. 3 (bottom left), as well as several other matrices from Tables 1 and 2, the p-CG and p-CG-rr methods show delayed convergence. Indeed, apart from the loss of attainable accuracy, the propagation of local rounding errors in many-recurrence variants of iterative schemes can cause a delay of convergence. We refer to [34], Section 5, and the references therein, in particular [18, 26, 28, 29] and [30] for a more detailed discussion on delay of convergence by loss of orthogonality due to roundoff. This delay translates into a larger number of iterations required to reach a certain accuracy, see Table 2. Although the residual replacement strategy ensures that a high accuracy can be obtained, it does not resolve the delay of convergence, as illustrated by the numerical results in this section. Hence, when application demands, a high accuracy can always be obtained using the p-CG-rr method, but may come at the cost of additional iterations, inducing a trade-off between accuracy and computational effort. The next section illustrates that a substantial speed-up can be obtained by using the pipelined methods.

4.3 Parallel performance

This section demonstrates that the parallel scalability of the pipelined CG method is maintained by the addition of the automated residual replacement strategy. The following parallel experiment is performed on a small cluster with 28 compute nodes, consisting of two 6-core Intel Xeon X5660 Nehalem 2.80 GHz processors each (12 cores per node). Nodes are connected by $4 \times$ QDR InfiniBand technology, providing 32 Gb/s of point-to-point bandwidth for message passing and I/O.

We use PETSc [1] version 3.6.3, which includes implementations of the CG and p-CG algorithms. Additionally, the p-CG-rr algorithm was implemented as a direct extension of p-CG. The benchmark problem used to asses strong scaling parallel performance is a moderately-sized 2D Poisson model, available in the PETSc distribution as example 2 in the Krylov solvers (KSP) folder. The simulation domain is discretized using a second order finite difference stencil with 1000×1000 grid points (1 million unknowns). No preconditioner is applied. The tolerance for Krylov solution imposed on the scaled recursive residual norm $\|r_i\|_2/\|b\|_2$ is 10^{-6}.

Since each node consists of 12 cores, we use 12 MPI processes per node to fully exploit parallelism on the machine. The MPI library used for this experiment is MPICH-3.1.3[2]. Note that the environment variables `MPICH_ASYNC_PROGRESS=1` and `MPICH_MAX_THREAD_SAFETY=multiple` are set to ensure optimal parallelism.

[2] `http://www.mpich.org/`

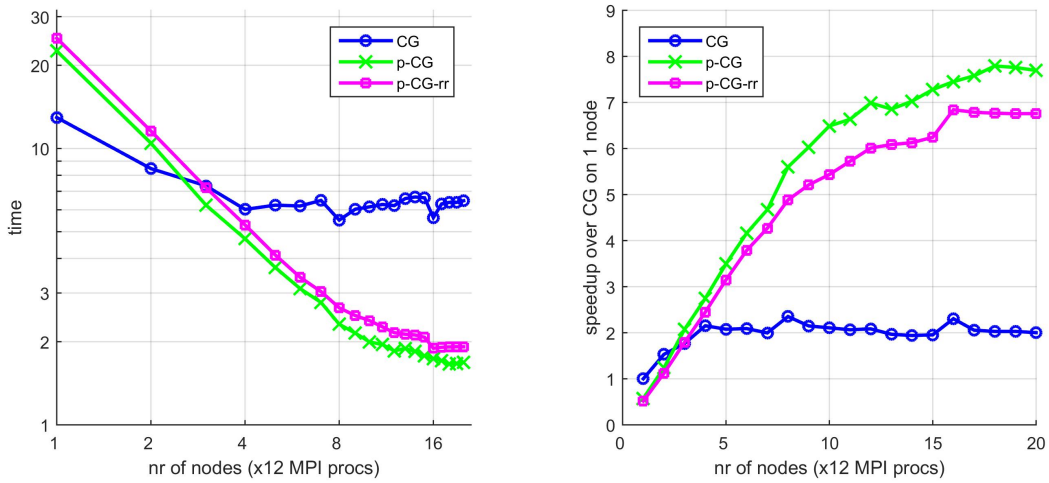


Figure 4: *Strong scaling experiment on up to* 20 *nodes* (240 *cores) for a 2D Poisson problem with* 1.000.000 *unknowns. Left: Absolute time to solution (*`log10` *scale) as function of number of nodes (*`log2` *scale). Right: Speedup over single-node classical CG. All methods converged in* 1474 *iterations to a relative residual tolerance* 1e-6*; p-CG-rr performed* 39 *replacements.*

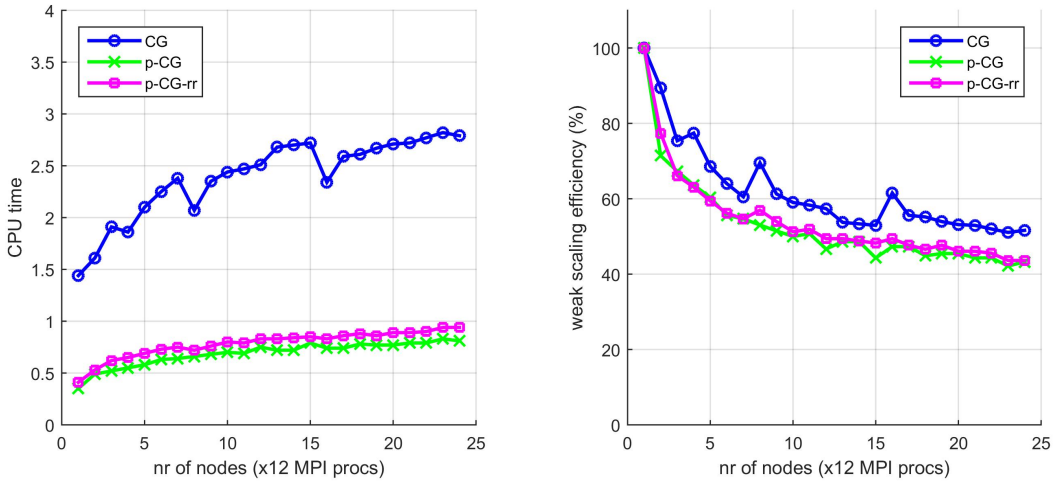


Figure 5: *Weak scaling experiment on up to* 24 *nodes* (288 *cores) for a 2D Poisson problem with* 62.500 *unknowns per node* (5200 *unknowns/core). Left: Absolute time* t_{CG} (600 *iterations) as function of number of nodes. Right: Weak scaling efficiency relative to single-node execution:* $\text{eff}_{CG}(m) = t_{CG}(1\text{ node})/t_{CG}(m\text{ nodes})$*. p-CG-rr performed* 10 *replacements.*

Figure 4 (left) shows the time to solution as a function of the number of nodes (strong scaling). Pipelined CG starts to outperform classical CG when the number of nodes exceeds two. Classical CG stops scaling from 4 nodes onward. The pipelined methods scale well on up to 20 nodes, see Fig. 4 (right). The maximum speed-up for p-CG on 20 nodes compared to CG on a single node is 7.7×, whereas the CG method achieves a speedup of only 2.0× on 20 nodes. This implies pipelined CG attains a net speedup of 3.8× over classical CG when both are executed on 20 nodes. Performance of p-CG-rr is comparable to that of p-CG. A minor slowdown is observed, which is partly due to the computation and communication of the additional norms that are required for the automated replacements, see (36)-(39), but primarily due to the additional computational work required for the SPMVs (52) when replacement takes place. The maximum speedup for p-CG-rr on 20 nodes compared to CG on a single node is 6.8×, implying a total speedup of 3.4× compared to standard CG on 20 nodes.

Figure 5 displays results for a weak scaling experiment, where the size of the Poisson problem grows linearly with respect to the number of cores. A fixed number of 62.500 unknowns per node is used. The problem hence consists of 1225×1225 (1.5 million) unknowns on 24 nodes. Fig. 5 (left) shows the time required to perform 600 iterations (fixed) of the various methods on up to 24 nodes. The speed-up observed for the pipelined methods in Fig. 4 is again visible here. The weak scaling efficiency of the p-CG and p-CG-rr algorithms (relative to their respective single-node execution) on 24 nodes (43%) is comparable to that of classical CG (51%), see Fig. 5 (right).

Figure 6 shows the accuracy of the solution in function of the number of iterations (left) and computational time (right) spent by the algorithms for the 2D Poisson 1000×1000 benchmark problem on a 20 node setup. In 3.2 seconds ($\sim$ 2500 iterations) the p-CG-rr algorithm obtains a solution with true residual norm 7.5e-12. Classical CG is over three times slower, requiring 11.1 seconds to attain a comparable accuracy (residual norm 9.4e-12), see also Fig. 4. The p-CG method without residual replacement is unable to reach a comparable accuracy regardless of computational effort. Indeed, stagnation of the true residual norm around 2.0e-7 is imminent from a total time of 2.0 seconds ($\sim$ 1800 iterations) onward. For completeness we note that the speed-up of p-CG/p-CG-rr over classical CG can also be obtained for less accurate final solutions, e.g. with $\|r_i\| = 10^{-8}$ or 10^{-6}, as shown by Fig. 6 (right).

5 Conclusions

The deviation of the recursively computed residuals from the true residuals due to the propagation of rounding errors is a well-known issue in many numerical methods. In particular for the Conjugate Gradients method, the true residuals typically stagnate under the influence of local roundoff errors, whereas the recursive residuals keep decreasing. This behavior is significantly more prominent in multi-recursion variants of CG, such as the three-term recurrence CG algorithm [33], Chronopoulos & Gear's communication avoiding version of CG [5], and the communication hiding pipelined CG method [17]. For these methods the possibly dramatic propagation of rounding errors may lead to a stagnation of the residual norm at several orders of magnitude above the accuracy attainable by classical CG, or may even result in a significant delay of convergence.

This paper analyzes the propagation of local rounding errors that stem from the recursions in classical CG, Chronopoulous & Gear's CG (CG-CG), and the pipelined CG (p-CG) algorithm. The preconditioned pipelined CG method features many additional recursively computed auxiliary variables compared to classical CG, which are all prone to rounding errors. It is shown that the gap between the explicitly computed and recursive residual is directly related (i.e. coupled)

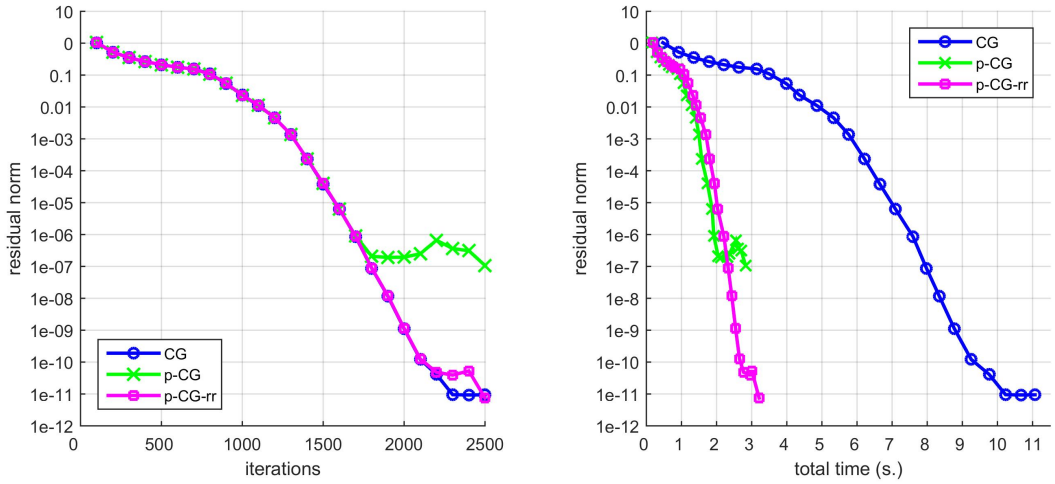


Figure 6: *Accuracy experiment on* 20 *nodes* (240 *cores) for a 2D Poisson problem with* 1.000.000 *unknowns. Left: Explicitly computed residual as function of iterations. Right: Residual as function of total time spent by the algorithm. Maximal number of iterations is* 2500 *for all methods; p-CG-rr performed (maximum)* 39 *replacements.*

to the gaps for the other recursively defined auxiliary variables, and is hence influenced by their respective local rounding errors. A bound on the residual gap norm, which explains the loss of accuracy in the pipelined CG algorithm, is derived.

We introduce a countermeasure to the rounding error propagation in pipelined CG in the form of an automated residual replacement strategy. The replacement criterion requires an accurate model for the residual gap in each iteration. A practically useable estimate for the residual gap is suggested based on the coupled system of recursions from the error analysis. The proposed model allows to predict the size of the error on the residual in each step of the pipelined algorithm. Whenever the estimated residual gap becomes too large relative to the residual, all auxiliary variables are computed explicitly to reset the build-up rounding errors.

The automated residual replacement strategy is validated on a variety of numerical benchmark problems. We show that the replacements allows to recover the accuracy of the classical CG method, which is unobtainable by pipelined CG. The incorporation of the replacement strategy in the pipelined CG algorithm requires the calculation of several additional vector norms; however, these computations can easily be combined with the existing global communication phase such that performance of the algorithm remains intact. Results with a parallel implementation of p-CG-rr in PETSc using the MPI message passing paradigm are presented to demonstrate that the replacement strategy improves accuracy, but does not impair parallel performance.

6 Acknowledgments

This work is funded by the EXA2CT European Project on Exascale Algorithms and Advanced Computational Techniques, which receives funding from the EU's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 610741. Additionally, S. C. is funded by the Research Foundation Flanders (FWO). The authors would like to cordially thank Zdeněk

Strakoš for his useful comments on earlier versions of this manuscript.

References

[1] S. Balay, S. Abhyankar, M.F. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, V. Eijkhout, W.D. Gropp, D. Kaushik, M.G. Knepley, L. Curfman McInnes, K. Rupp, B.F. Smith, S. Zampini, and H. Zhang. PETSc Web page. `http://www.mcs.anl.gov/petsc`, 2015.

[2] R. Barrett, M. Berry, T.F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H.A. Van der Vorst. Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods. *2nd ed., SIAM, Philadelphia*, 1994.

[3] E. Carson and J. Demmel. A residual replacement strategy for improving the maximum attainable accuracy of s-step Krylov subspace methods. *SIAM Journal on Matrix Analysis and Applications*, 35(1):22–43, 2014.

[4] E. Carson, N. Knight, and J. Demmel. Avoiding communication in nonsymmetric Lanczos-based Krylov subspace methods. *SIAM Journal on Scientific Computing*, 35(5):S42–S61, 2013.

[5] A.T. Chronopoulos and C.W. Gear. s-Step iterative methods for symmetric linear systems. *Journal of Computational and Applied Mathematics*, 25(2):153–168, 1989.

[6] A.T. Chronopoulos and A.B. Kucherov. Block s-step Krylov iterative methods. *Numerical Linear Algebra with Applications*, 17(1):3–15, 2010.

[7] A.T. Chronopoulos and C.D. Swanson. Parallel iterative s-step methods for unsymmetric linear systems. *Parallel Computing*, 22(5):623–641, 1996.

[8] E. De Sturler. A parallel variant of GMRES(m). In *Proceedings of the 13th IMACS World Congress on Computational and Applied Mathematics*, volume 9, 1991.

[9] E. De Sturler and H.A. Van der Vorst. Reducing the effect of global communication in GMRES(m) and CG on parallel distributed memory computers. *Applied Numerical Mathematics*, 18(4):441–459, 1995.

[10] J.W. Demmel. *Applied Numerical Linear Algebra*. SIAM, 1997.

[11] J.W. Demmel, M.T. Heath, and H.A. Van der Vorst. Parallel Numerical Linear Algebra. *Acta Numerica*, 2:111–197, 1993.

[12] J. Dongarra and M.A. Heroux. Toward a new metric for ranking high performance computing systems. *Sandia National Laboratories Technical Report, SAND2013-4744*, 312, 2013.

[13] J. Dongarra, M.A. Heroux, and P. Luszczek. HPCG benchmark: a new metric for ranking high performance computing systems. *University of Tennessee, Electrical Engineering and Computer Sciente Department, Technical Report UT-EECS-15-736*, 2015.

[14] P.R. Eller and W. Gropp. Non-blocking preconditioned Conjugate Gradient methods for extreme-scale computing. In *Conference proceedings*. 17th Copper Mountain Conference on Multigrid Methods, Colorado, US, 2015.

[15] J. Erhel. A parallel GMRES version for general sparse matrices. *Electronic Transactions on Numerical Analysis*, 3(12):160–176, 1995.

[16] P. Ghysels, T.J. Ashby, K. Meerbergen, and W. Vanroose. Hiding global communication latency in the GMRES algorithm on massively parallel machines. *SIAM Journal on Scientific Computing*, 35(1):C48–C71, 2013.

[17] P. Ghysels and W. Vanroose. Hiding global synchronization latency in the preconditioned Conjugate Gradient algorithm. *Parallel Computing*, 40(7):224–238, 2014.

[18] A. Greenbaum. Behavior of slightly perturbed Lanczos and Conjugate-Gradient recurrences. *Linear Algebra and its Applications*, 113:7–63, 1989.

[19] A. Greenbaum. Estimating the attainable accuracy of recursively computed residual methods. *SIAM Journal on Matrix Analysis and Applications*, 18(3):535–551, 1997.

[20] M.H. Gutknecht and Z. Strakoš. Accuracy of two three-term and three two-term recurrences for Krylov space solvers. *SIAM Journal on Matrix Analysis and Applications*, 22(1):213–229, 2000.

[21] N. Halko, P.G. Martinsson, and J.A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288, 2011.

[22] M.R. Hestenes and E. Stiefel. Methods of Conjugate Gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 14(6), 1952.

[23] N.J. Higham. *Accuracy and stability of numerical algorithms*. SIAM, 2002.

[24] J. Liesen and Z. Strakoš. *Krylov Subspace Methods: Principles and Analysis*. Oxford University Press, 2012.

[25] G. Meurant and Z. Strakoš. The Lanczos and Conjugate Gradient algorithms in finite precision arithmetic. *Acta Numerica*, 15:471–542, 2006.

[26] Y. Notay. On the convergence rate of the Conjugate Gradients in presence of rounding errors. *Numerische Mathematik*, 65(1):301–317, 1993.

[27] C.C. Paige. *The computation of eigenvalues and eigenvectors of very large sparse matrices*. PhD thesis, University of London, 1971.

[28] C.C. Paige. Computational variants of the Lanczos method for the eigenproblem. *IMA Journal of Applied Mathematics*, 10(3):373–381, 1972.

[29] C.C. Paige. Error analysis of the Lanczos algorithm for tridiagonalizing a symmetric matrix. *IMA Journal of Applied Mathematics*, 18(3):341–349, 1976.

[30] C.C. Paige. Accuracy and effectiveness of the Lanczos algorithm for the symmetric eigenproblem. *Linear Algebra and its Applications*, 34:235–258, 1980.

[31] G.L.G. Sleijpen and H.A. Van der Vorst. Reliable updated residuals in hybrid Bi-CG methods. *Computing*, 56(2):141–163, 1996.

[32] G.L.G. Sleijpen, H.A. Van der Vorst, and J. Modersitzki. Differences in the effects of rounding errors in Krylov solvers for symmetric indefinite linear systems. *SIAM Journal on Matrix Analysis and Applications*, 22(3):726–751, 2001.

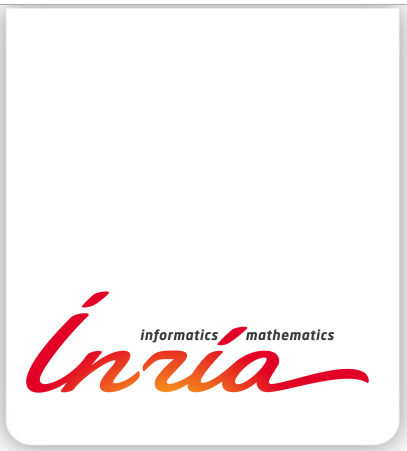
[33] E. Stiefel. Relaxationsmethoden bester Strategie zur Lösung linearer Gleichungssysteme. *Comm. Math. Helv.*, 29(1):157–179, 1955.

[34] Z. Strakoš and P. Tichý. On error estimation in the Conjugate Gradient method and why it works in finite precision computations. *Electronic Transactions on Numerical Analysis*, 13:56–80, 2002.

[35] C. Tong and Q. Ye. Analysis of the finite precision Bi-Conjugate Gradient algorithm for nonsymmetric linear systems. *Mathematics of Computation*, 69(232):1559–1575, 2000.

[36] H.A. Van der Vorst and Q. Ye. Residual replacement strategies for Krylov subspace iterative methods for the convergence of true residuals. *SIAM Journal on Scientific Computing*, 22(3):835–852, 2000.

[37] J.H. Wilkinson. *Rounding errors in algebraic processes*. Courier Corporation, 1994.



RESEARCH CENTRE
BORDEAUX – SUD-OUEST

200, Avenue de la vieille tour
33405 Talence Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399